Faktorizace

# Úvodní informace

V této úloze budeme rozkládat přirozená čísla na součin prvočinitelů. Jedná se o matematický postup, který se vyučuje na druhém stupni ZŠ. Princip by studenti tudíž měli znát. Příklad se svojí obtížností zařazuje mezi ty těžké. Složitější je si uvědomit, jak přesně aplikovat princip rozkladu, aby byl univerzálně použitelný na každé přirozené číslo.

# Cíle úlohy

* Procvičení:
  + Cykly
  + Vnořené cykly
  + Funkce vlastní

# Náročnost

* 1-2 vyučovací hodiny
* Obtížnost:

# Prerekvizity

* Cykly
* Vnořené cykly
* Funkce vlastní
* Znalost definice prvočísla

# Metodika výuky

Jak již bylo zmíněno v úvodních informacích, tak tato úloha by měla být principiálně známá všem studentům, proto jim můžeme hned na začátku představit zadání a až poté vysvětlíme samotný algoritmus.

Zadání:

Vymyslete algoritmus pro faktorizaci složených čísel. Vstupem bude přirozené číslo, které bude následně buď rozloženo na prvočísla, nebo v případě, kdy se jedná o prvočíslo, vypíše jako jediného dělitele zadané číslo. Všechny prvočinitele ukládejte do pole. Pokud programovací jazyk potřebuje znát velikost pole, udejte velikost 10000. Samotnou faktorizaci bude provádět (včetně výpisu) vlastní fuknce *faktorizace*.

Vstup: 2772

Výstup: 2, 2, 3, 3, 7, 11

Vstup: 73

Výstup: 73

V první řadě je potřeba provést rozbor algoritmu. Můžeme jej hned zahájit diskusí (Otázky do diskuse – rozbor algoritmu).

### Otázky do diskuse – rozbor algoritmu

1. Jak bude vypadat algoritmus pro faktorizaci? Jaký by mohl být první dělitel čísla 2772?
2. Jak budeme postupovat dál? Kterého dělitele budeme testovat jako dalšího?

Jak bude vypadat algoritmus pro faktorizaci čísla N? Pro zadané číslo N budeme postupně testovat všechny dělitele D z intervalu od 2 do N - 1.

1. Začneme s D = 2
2. Je-li D větší než N, algoritmus končí
3. Zjistíme, zda je D dělitelem N
   1. v případě, že ano, uložíme D do pole, změníme N na podíl N/D (čili: N = N/D) a pokračujeme bodem 2
   2. v případě, že ne, zvětšíme D o 1 a pokračujeme bodem 2

Pro samotné vysvětlení můžeme zvolit pomocný slide, konkrétně faktorizace.png.

Obsah obrázku text, dopis

Popis byl vytvořen automaticky

Obrázek 1 – Vysvětlení na rozkladu čísla 2772

Je vhodné také zmínit záludnost, kdy by se mohlo zdát, že bude stačit testovat pouze prvočísla (jelikož výsledkem rozkladu složeného čísla jsou vždy prvočinitelé). To je sice pravda, ale takový seznam by musel být již předdefinovaný (například v poli). Vytváření vlastní tabulky prvočísel, či samotné testování prvočíselnosti při faktorizaci, je zcela neefektivní a navíc složité.

Příklad je možné řešit samostatně i společně. U individuální práce dbáme na to, aby všichni studenti rozuměli algoritmu. Při zpracování může nastat problém u striktních programovacích jazyků, jako je například C++, kdy je nutné zadat při deklaraci velikost pole. Tendence studentů bude tam „naprat“ velké číslo a tím problém obejít (v podstatě jsme jim to zadali i v samotném zadání). Vhodné metody, jak dosáhnout nejmenší možné velikosti pole, jsou popsány v následné diskusi, která by po splnění zadání měla proběhnout. Nyní se již mohou studenti pustit do psaní algoritmu. Součástí vzorového řešení je program faktorizace. Po úspěšném naprogramování je vhodné program promítnout a studentům případné nejasnosti vysvětlit.

### Otázky do diskuse – reflexe faktorizace

1. Pokud bychom programovali v striktním jazyku jako například C++, kde musíme zadat při deklaraci velikost pole, jak byste postupovali, aby byla velikost pole co možná nejmenší? (vysvětlení níže)
2. Proč je důležité nenastavovat velikost pole „natvrdo“?
3. Co vás na úloze bavilo?

Vysvětlení otázky č.1:

Existují v podstatě dva různé způsoby řešení, jak určit dostatečnou velikost pole:

1. Vyškrtneme ze zadání pasáž ukládání prvočinitelů do pole. Studenti ho při nalezení ihned vypíší. Tím tento problém eliminujeme, ale nejedná se o preferovanou možnost, navíc v zadání otázky se s polem počítá.
2. Vypočítat, jaký nejvyšší možný počet prvočinitelů mají jednotlivá čísla. To lze zjistit snadno, ale neintuitivně. Platí, že každé číslo X, které je mocninou dvojky (2, 4, 8, 16, …), má větší počet prvočinitelů, než kterékoli menší číslo (viz obrázek níže). Stačí tedy pro zadané N najít nejbližší vyšší X a protože jeho prvočinitele jsou jen dvojky, snadno určíme jejich počet. Velikost pole pak bude počet-1.

Obsah obrázku diagram, text

Popis byl vytvořen automaticky

Obrázek 2 - Maximální počet prvočinitelů

Pokud nám to časová dotace dovolí, tak si můžeme řešení s maximální velikostí pole implementovat do již hotových příkladů. Pro to vytvoříme jednoduchý cyklus, který je součástí výsledného programu v jazyce C++ faktorizace-v2.cpp.

### Možné problémy

* Student nepochopí algoritmus*.* – Vrátíme se zpět k demonstrativnímu příkladu, ale zvolíme číslo sudé a malé (například 36) a necháme ho rozložit na výsledek stejným způsobem.
* Student nenajde všechny prvočíselné dělitele*.* – Pravděpodobně bude chybět v programu vnořený cyklus, který bude testovat číslo (a následné výsledky po dělení) do té doby, než dělitelem nebude.